APPLICATIONS OF MICROCONTROLLER(8051)

OUTLINE

- Where can we use?
- Display app's
- Real world components interfacing
- Motion control app's

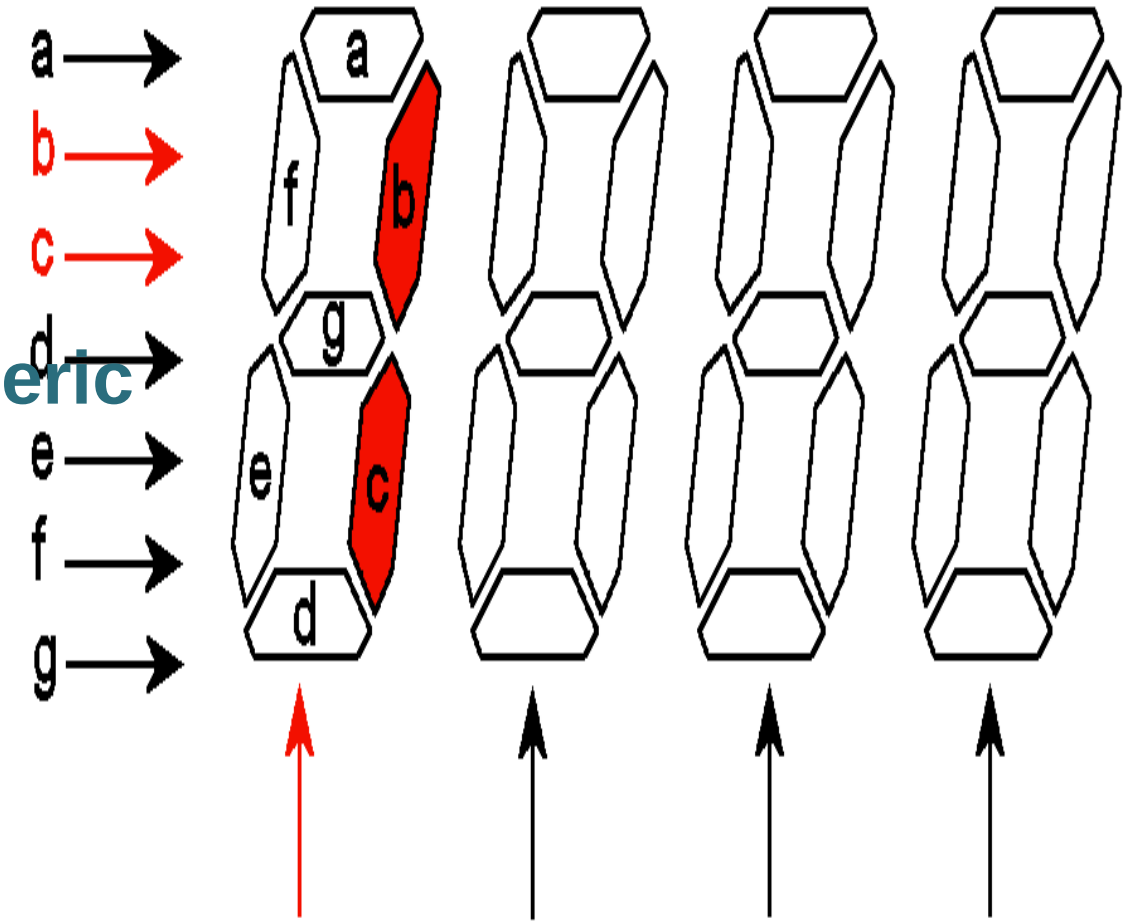
Where can we use?

Microcontrollers are mostly used in following electronic equipments

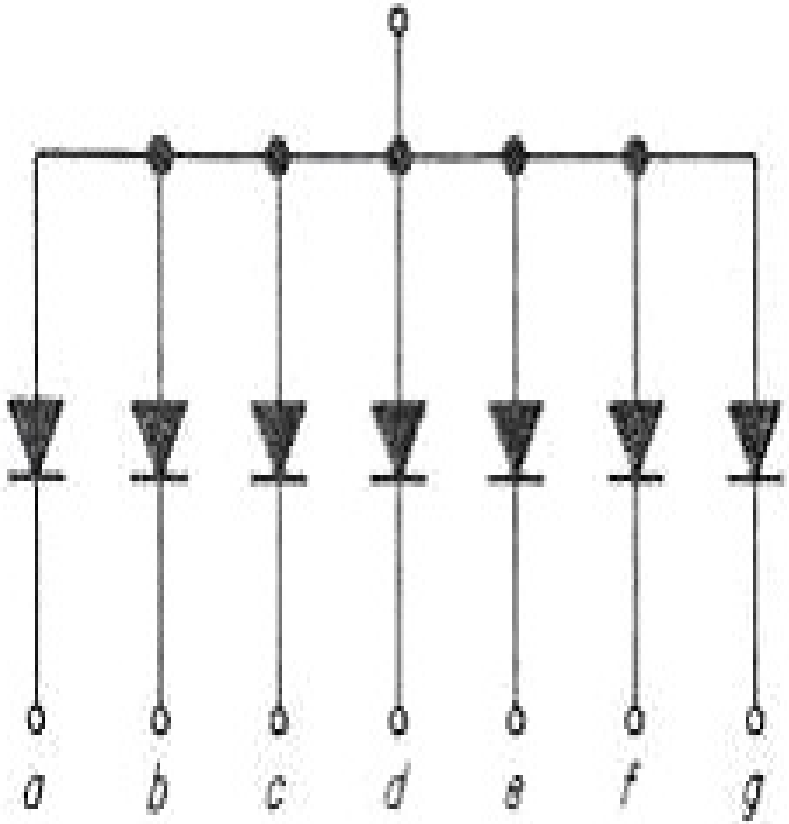
- Mobile Phones
- Auto Mobiles
- CD/DVD Players
- Washing Machines
- Cameras
- In Computers-> Modems and Keyboard Controllers
- Security Alarms
- Electronic Measurement Instruments
- Microwave Oven
- Automation And Control
- Stand Alone Control System

Display app's

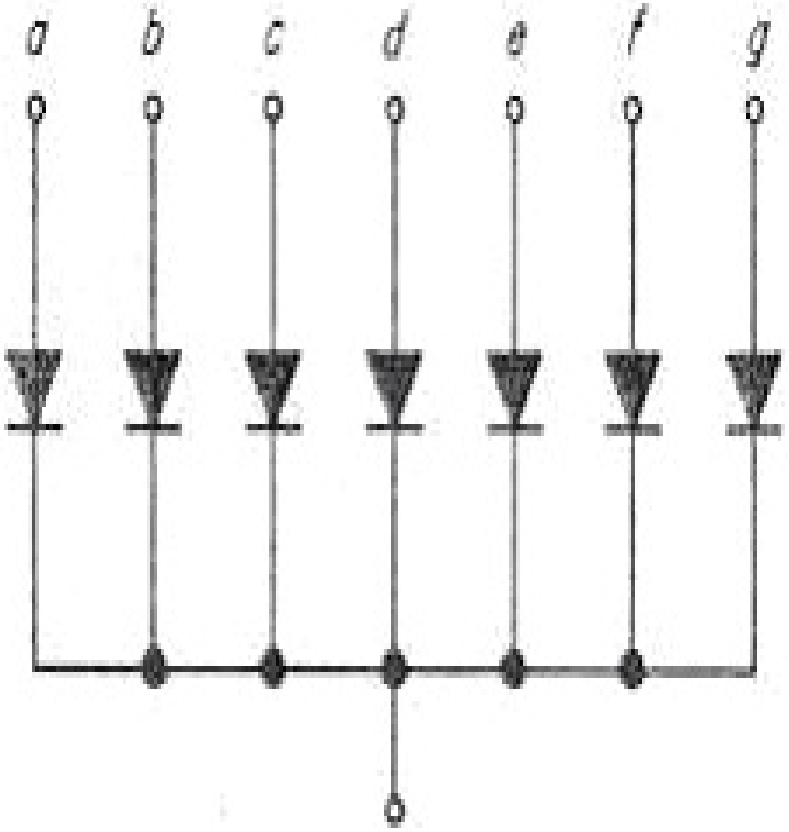
- Single light(s)
- Single characters
- Intelligent alphanumeric



Seven Segment Numeric Display

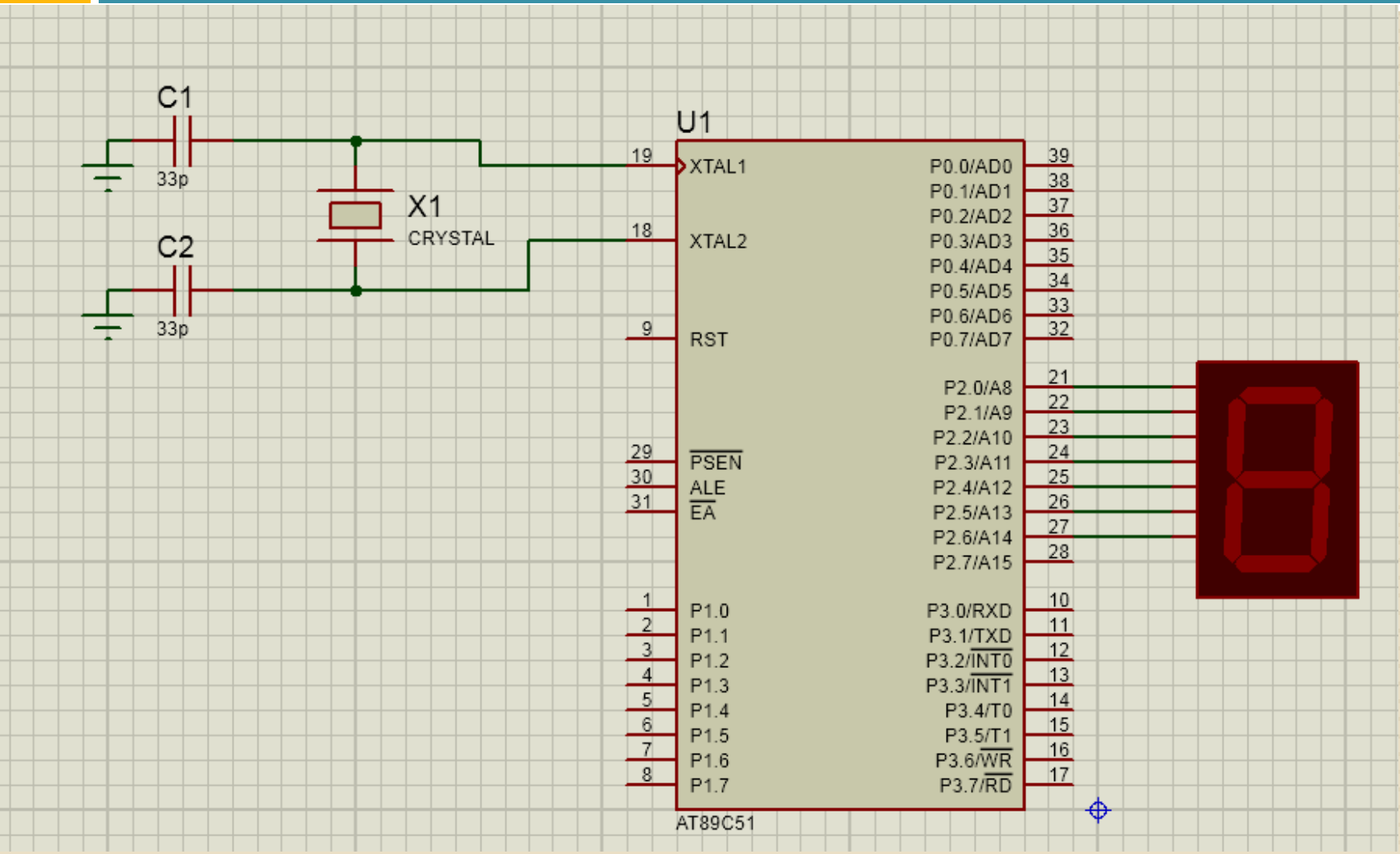


Common Anode



Common Cathode

7segment Led Interface To 8051-1



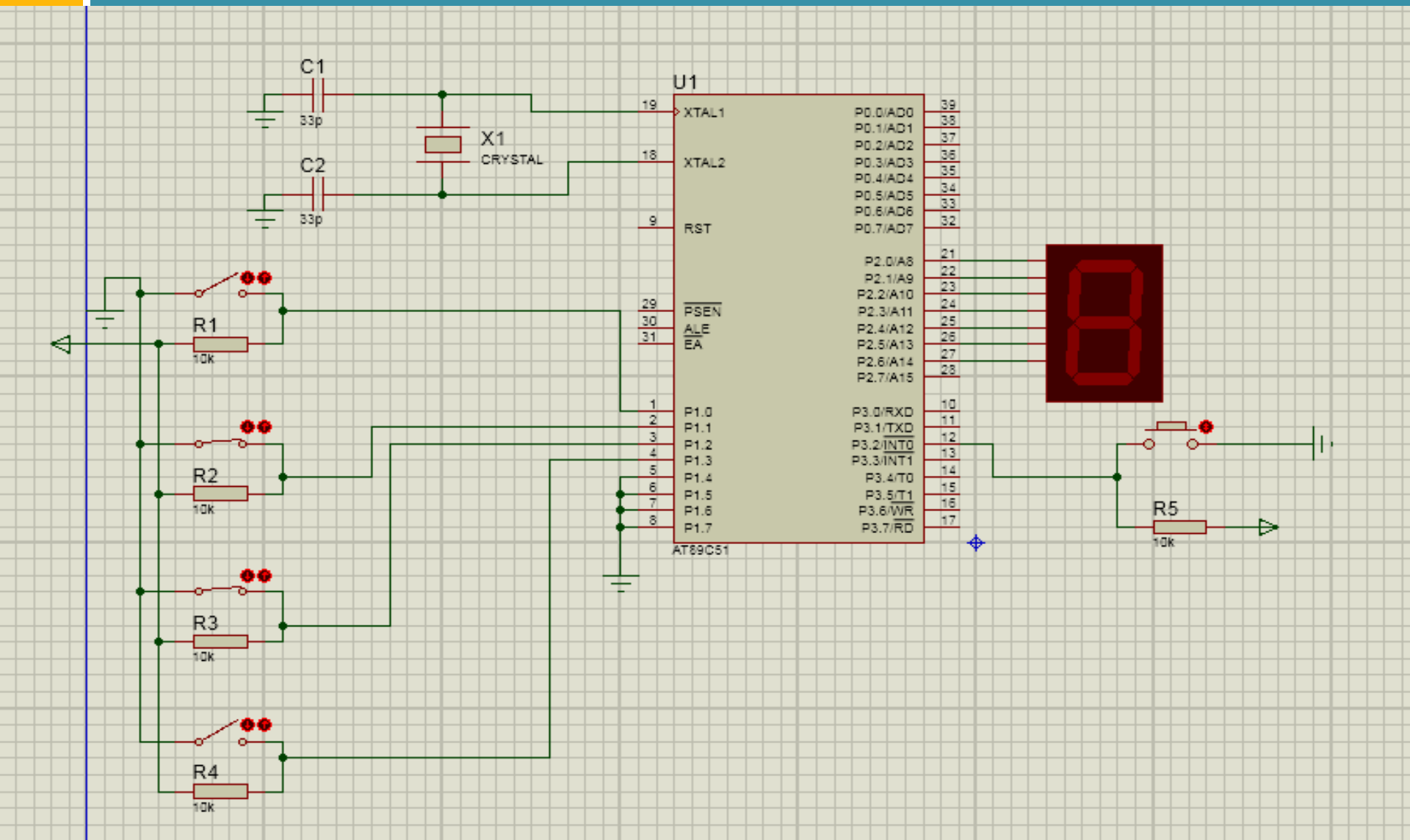
LOOK UP TABLE

DIGIT	g	f	e	d	c	b	a	HEX
0	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**3F**
1	0	0	0	0	1	1	0	06
2	1	0	1	1	0	1	1	5B
3	1	0	0	1	1	1	1	4F
4	1	1	0	0	1	0	0	66
5	1	1	0	1	1	0	1	6D
6	1	1	1	1	1	0	1	7D
7	0	0	0	0	1	1	1	07
8	1	1	1	1	1	1	1	7F
9	1	1	0	0	1	1	1	67
A	1	1	1	0	1	1	1	77
B	1	1	1	1	1	0	0	7C
C	0	1	1	1	0	0	1	39
D	1	0	1	1	1	1	0	5E
E	1	1	1	1	0	0	1	79
F	1	1	1	0	0	0	1	71

Write a program to continuously display 0-F on LED's connected to p2

```
ORG 0H
            LJMP    MAIN
            ORG     300H; initialization of memory
   MAIN:
   BACK1:   MOV     DPTR,#MYDATA; get the starting address of the look up table
            MOV     R5,#10H; count for the 16 digits
   BACK:    MOVC    A,@A+DPTR; move the digit to acc
            MOV     P2,A; display the digit
            INC     DPTR; get the next digit
            CLR     A
            ACALL   DELAY
            ACALL   DELAY
            ACALL   DELAY
            DJNZ    R5,BACK
            SJMP    BACK1
   DELAY:   MOV     TMOD,#01H
            MOV     TL0,#00
            MOV     TH0,#00
            SETB    TR0
   HERE:    JNB     TF0,HERE
            CLR     TR0
            CLR     TF0
            RET
ORG 40H
   MYDATA: DB 3FH,06H,5BH,4FH,66H,6DH,7DH,07H,7FH,67H,77H,7CH,39H,5EH,79H,71H
END
```

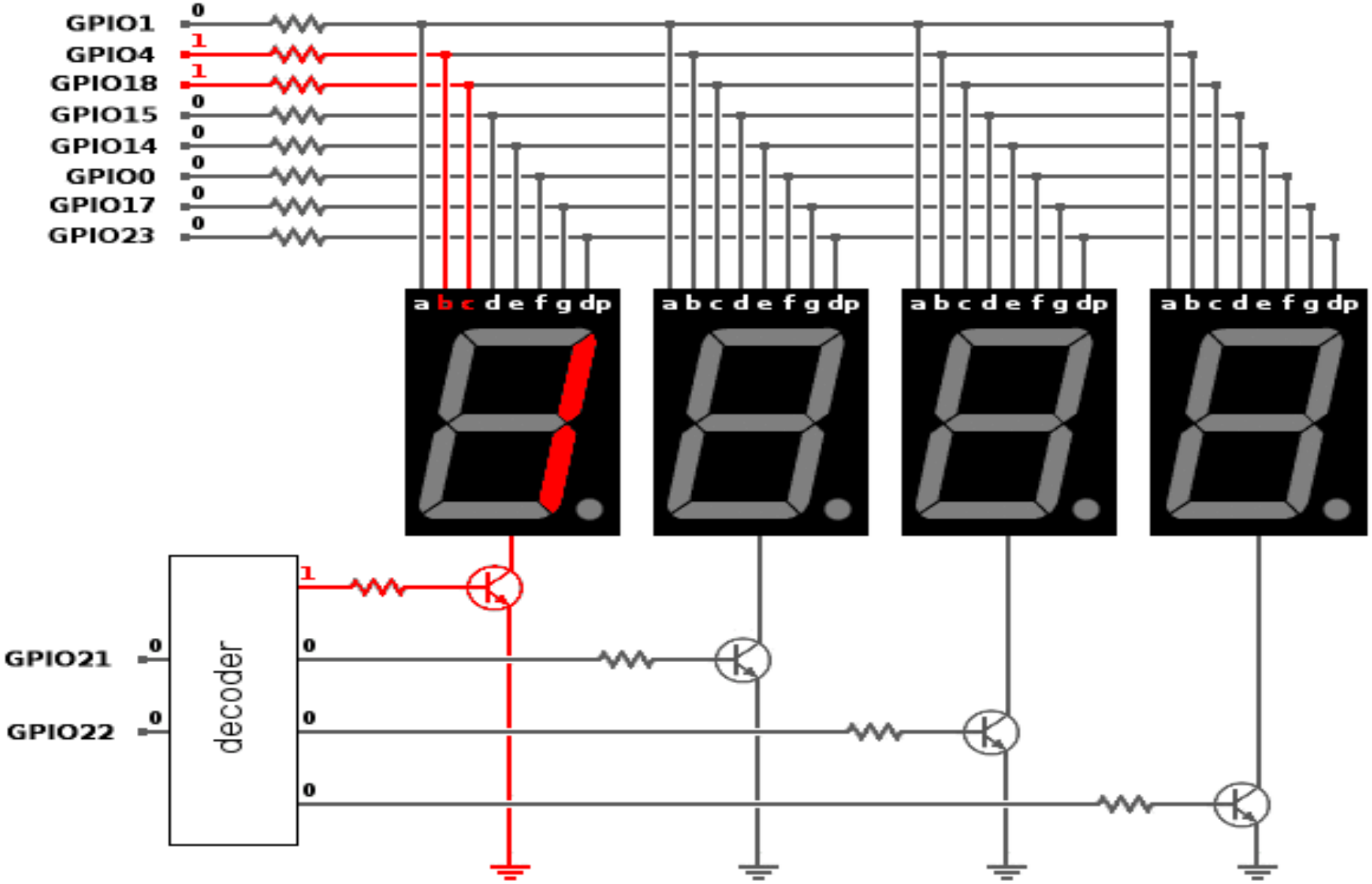
7segment Led Interface To 8051 -2



Write a program to continuously display 0-F and the value at P1 whenever an INT0 is generated.

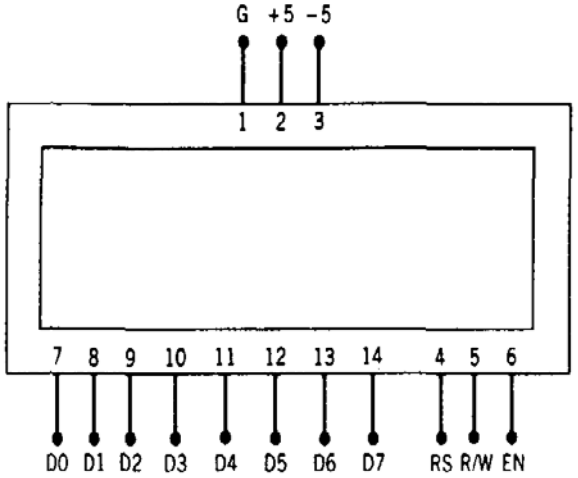
```
        ORG 0H
                LJMP    MAIN
                ORG     30H; initialization of memory
MAIN:
BACK1:  MOV     IE,#10000001B
        MOV     TCON,#00000010B
        MOV     DPTR,#MYDATA
        MOV     R6,#10H
BACK:   MOVC    A,@A+DPTR
        MOV     P2,A
        INC     DPTR
        CLR     A
        ACALL   DELAY
        DJNZ    R6,BACK
        SJMP    BACK1
DELAY:  MOV     R5,#05H
LAST:   MOV     TMOD,#01H
        MOV     TL0,#00
        MOV     TH0,#00
        SETB    TR0
HERE:   JNB     TF0,HERE
        CLR     TR0
        CLR     TF0
        DJNZ    R5,LAST
        RET

ORG 70H
MYDATA:DB
3FH,06H,5BH,4FH,66H,6DH,7DH,07H,7FH,67H,77H,
7CH,39H,5EH,79H,71H

ORG 003H
        LJMP    INT
ORG 90H
INT:    PUSH    DPL
        PUSH    DPH
        MOV     DPTR,#MYDATA
        MOV     R4,P1
Y1:     INC     DPTR
        DJNZ    R4,Y1
S:      CLR     A
        MOVC    A,@A+DPTR
        MOV     P2,A
        CLR     A
        POP     DPH
        POP     DPL
        RETI

END
```

Seven Segment Multiplexing using 8051



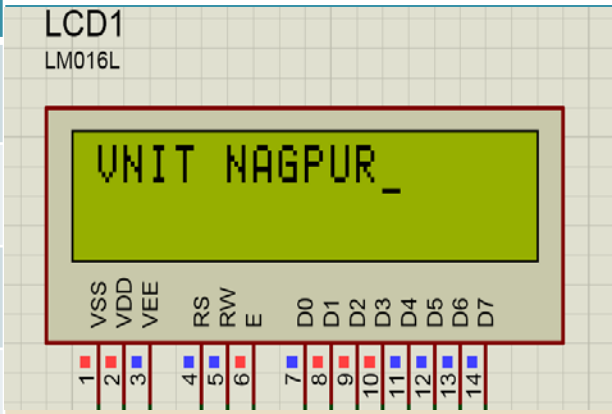
LCD Interfacing

Pin	Symbol	Description
1	VSS	Ground
2	VCC	Power supply
3	RS	Register select
4	VEE	Control contrast
5	R/W	Read/write
6	E	enable
7-14	DB0-DB7	8-bit data bus

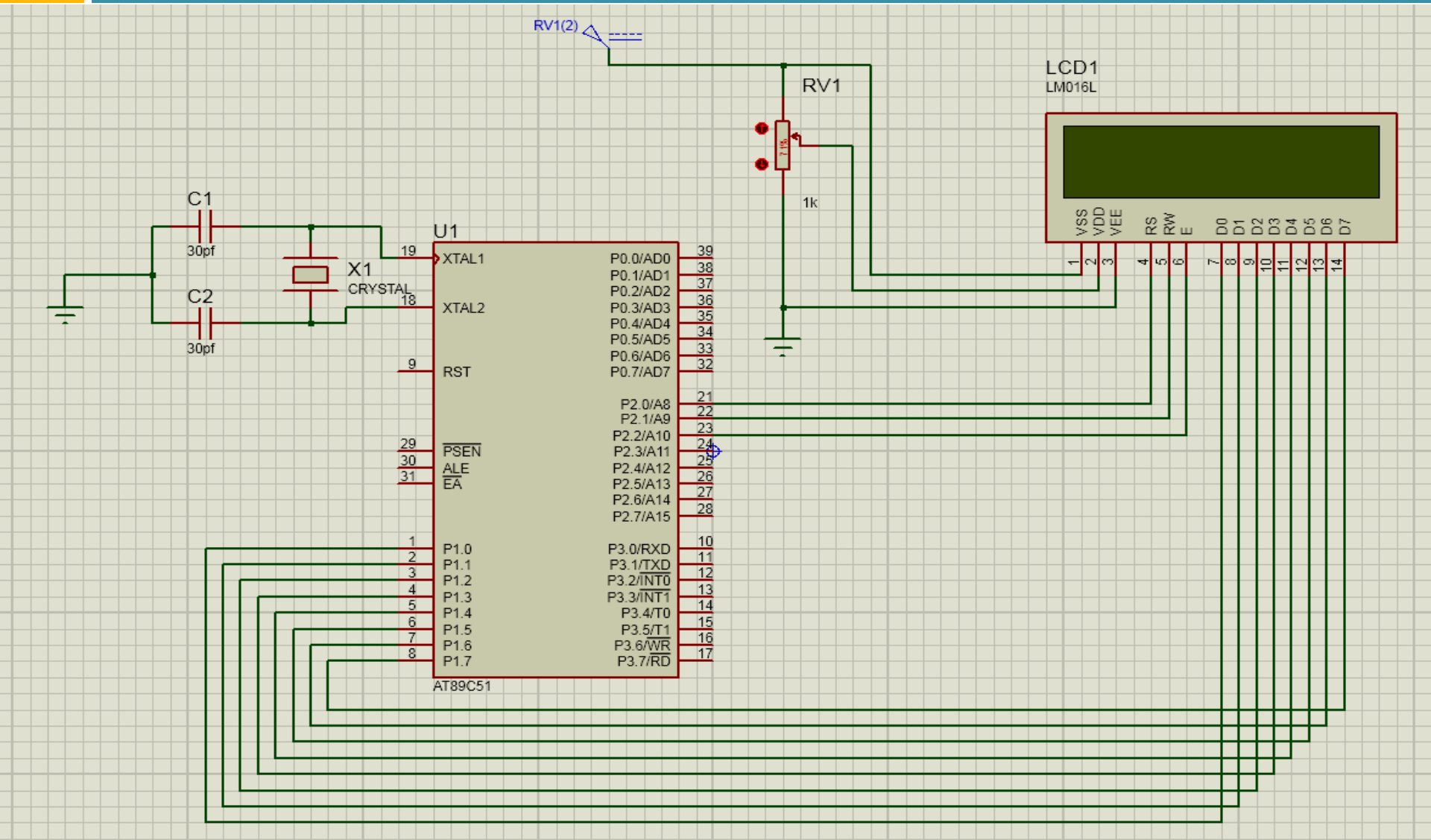


LCD command codes

Code (Hex)	Command to LCD Instruction Register	Code (Hex)	Command to LCD Instruction Register
1	Clear display screen	E	Display on, cursor blinking
2	Return home	F	Display on, cursor blinking
4	shift cursor to left	10	Shift cursor position to left
6	shift cursor to right	14	Shift cursor position to right
5	Shift display right	18	Shift the entire display to the left
7	Shift display left	1C	Shift the entire display to the right
8	Display off, cursor off	80	Force cursor to beginning to 1st line
A	Display off, cursor on	C0	Force cursor to beginning to 2nd line
C	Display on, cursor off	38	2 lines and 5x7 matrix



LCD Interfacing Circuit



•DISPLAY PROGRAM 1

```
        ORG 0H
                MOV     A,#38H          ;INIT. LCD 2 LINES, 5X7 MATRIX
                ACALL   COMNWRT         ;call command subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#0EH          ;display on, cursor on
                ACALL   COMNWRT         ;call command subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#01           ;clear LCD
                ACALL   COMNWRT         ;call command subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#06H          ;shift cursor right
                ACALL   COMNWRT         ;call command subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#84H          ;cursor at line 1, pos. 4
                ACALL   COMNWRT         ;call command subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#'N'          ;display letter N
                ACALL   DATAWRT         ;call display subroutine
                ACALL   DELAY           ;give LCD some time
                MOV     A,#'O'          ;display letter O
                ACALL   DATAWRT         ;call display subroutine
AGAIN:          SJMP    AGAIN           ;stay here
```

•DISPLAY PROGRAM 1 CONTD….

```
        COMNWRT:        ;send command to LCD
        MOV P1,A        ;copy reg A to port 1
        CLR P2.0        ;RS=0 for command
        CLR P2.1        ;R/W=0 for write
        SETB P2.2       ;E=1 for high pulse
        ACALL DELAY     ;give LCD some time
        CLR P2.2        ;E=0 for H-to-L pulse
    RET
        DATAWRT:                ;write data to LCD
        MOV P1,A        ;copy reg A to port 1
        SETB P2.0       ;RS=1 for data
        CLR P2.1        ;R/W=0 for write
        SETB P2.2       ;E=1 for high pulse
        ACALL DELAY     ;give LCD some time
        CLR P2.2        ;E=0 for H-to-L pulse
    RET
DELAY:  MOV R3,#50      ;50 or higher for fast CPUs
HERE2:  MOV R4,#255     ;R4 = 255
HERE:   DJNZ R4,HERE    ;stay until R4 becomes 0
        DJNZ R3,HERE2
RET
END
```

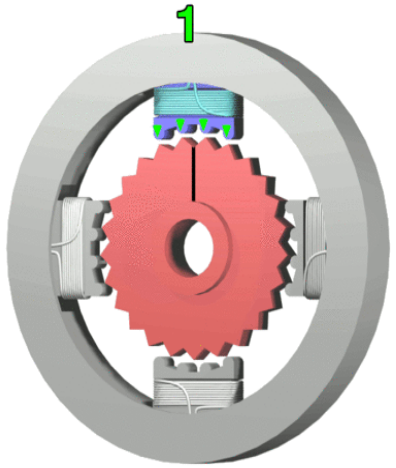
•DISPLAY PROGRAM 2

```
        ORG 0H
                MOV     A,#38H
                ACALL   COMMAND
                MOV     A,#0EH
                ACALL    COMMAND
                MOV     A,#01H
                ACALL   COMMAND
                MOV     A,#06H
                ACALL   COMMAND
                MOV     A,#86H
                ACALL   COMMAND
                MOV     A,#'V'
                ACALL   DATA_DISPLAY
                MOV     A,#'N'
                ACALL   DATA_DISPLAY
                MOV     A,#'I'
                ACALL    DATA_DISPLAY
                MOV     A,#'T'
                ACALL   DATA_DISPLAY
HERE:           SJMP    HERE
```

•DISPLAY PROGRAM 2 CONTD…

```
COMMAND:
        ACALL   READY
        MOV     P1,A
        CLR     P2.0
        CLR     P2.1
        SETB    P2.2
        CLR     P2.2
  RET
        DATA_DISPLAY:
        ACALL   READY
        MOV     P1,A
        SETB    P2.0
        CLR     P2.1
        SETB    P2.2
        CLR     P2.2
  RET
        READY:
        SETB    P1.7
        CLR     P2.0
        SETB    P2.1
        BACK:   SETB P2.2
        CLR     P2.2
        JB      P1.7,BACK
        RET
  END
```

Stepper Motor Interfacing



APPLICATIONS

- DISK DRIVES
  - DOT MATRIX PRINTER
    - ROBOTICS
      - POSITION CONTROL

CLASSIFICATION

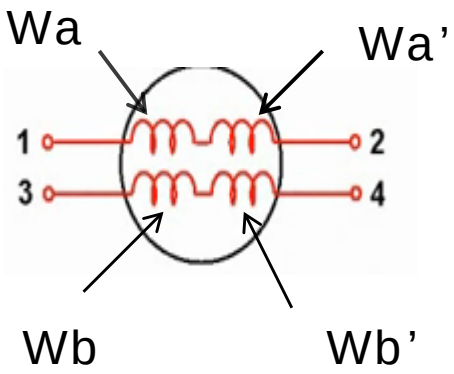
CONSTRUCTION

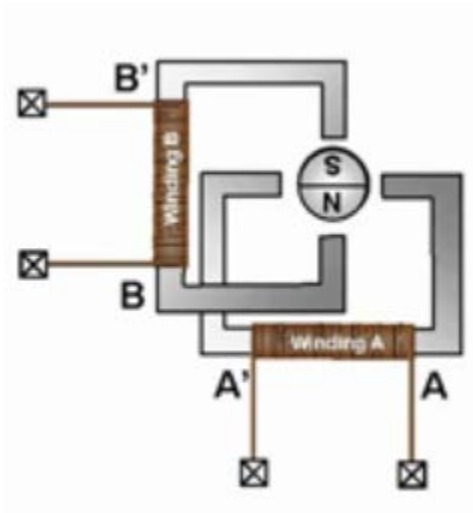
- Unipolar
- Bipolar

OPERATION

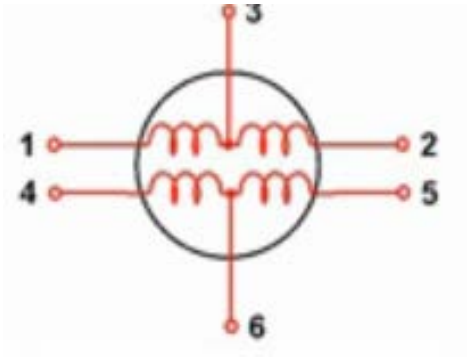
- Full step excited(FSE)
- Dual phase full step excited(DFSE)
- Half step excited(HSE)

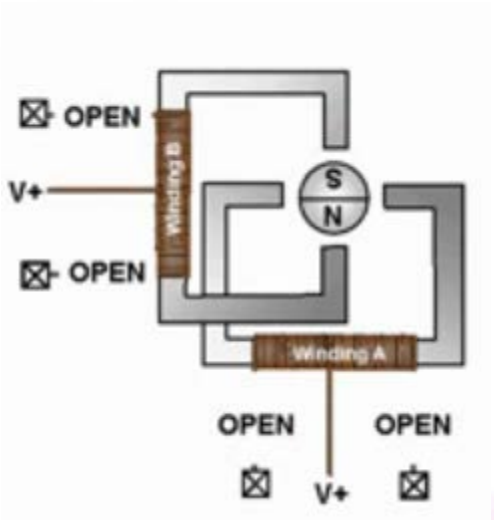
BIPOLAR



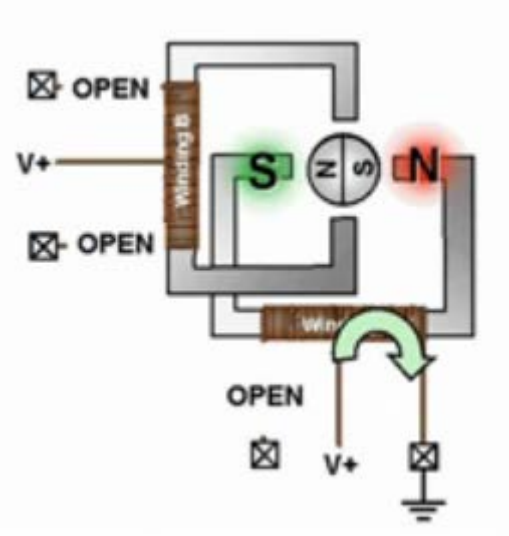


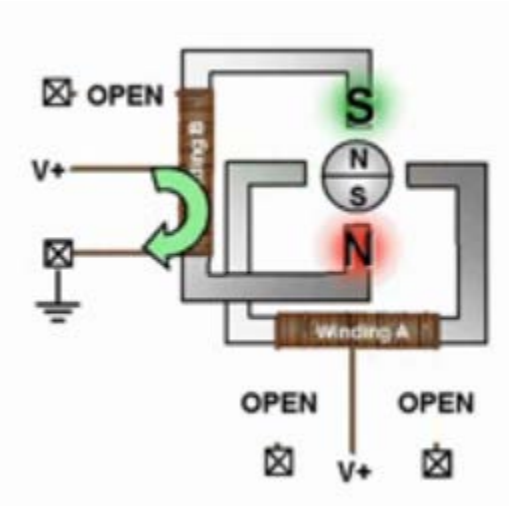
UNIPOLAR

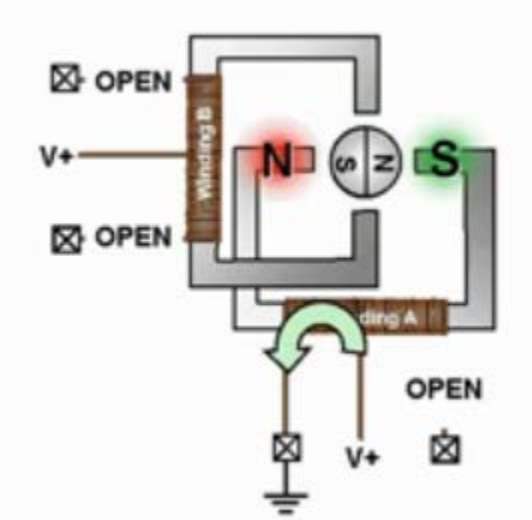


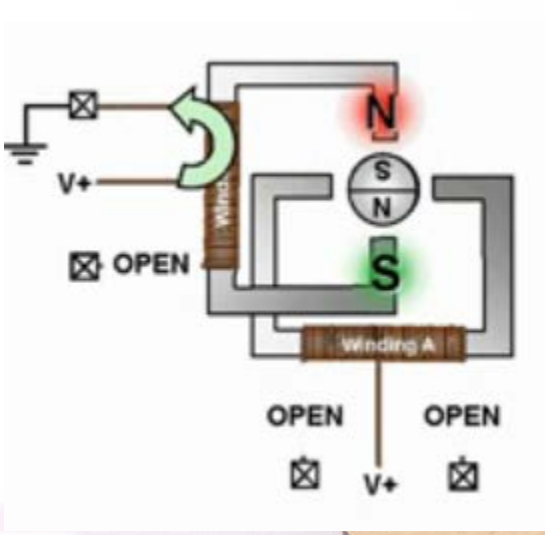


Unipolar

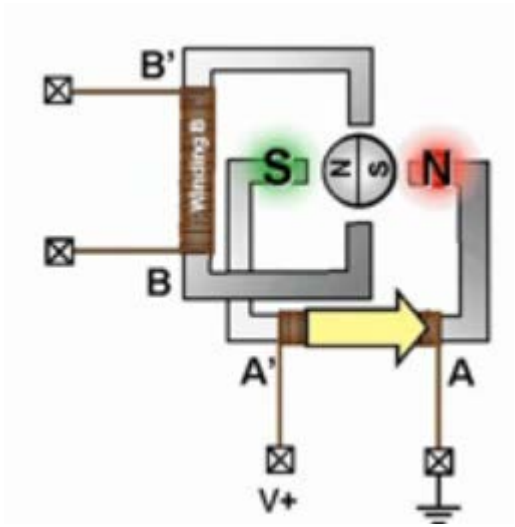

OPEN
V+
OPEN
S
N
OPEN
V+

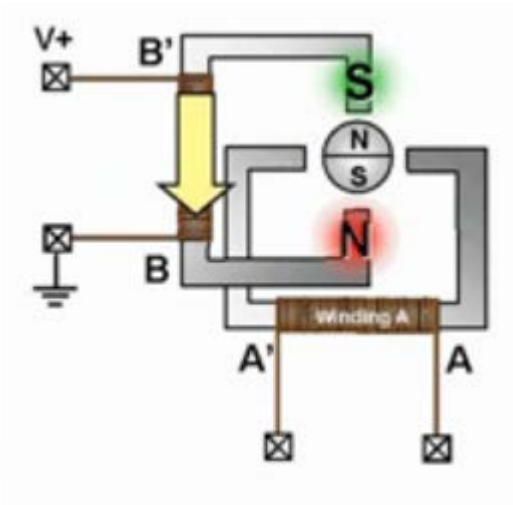

OPEN
V+
S
N
N
OPEN
OPEN
V+

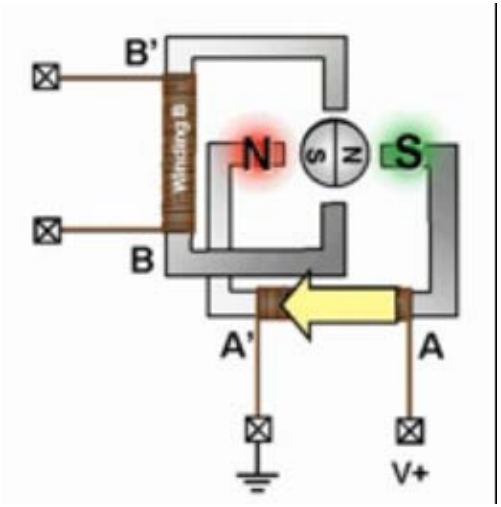

OPEN
V+
OPEN
N
S
OPEN
V+

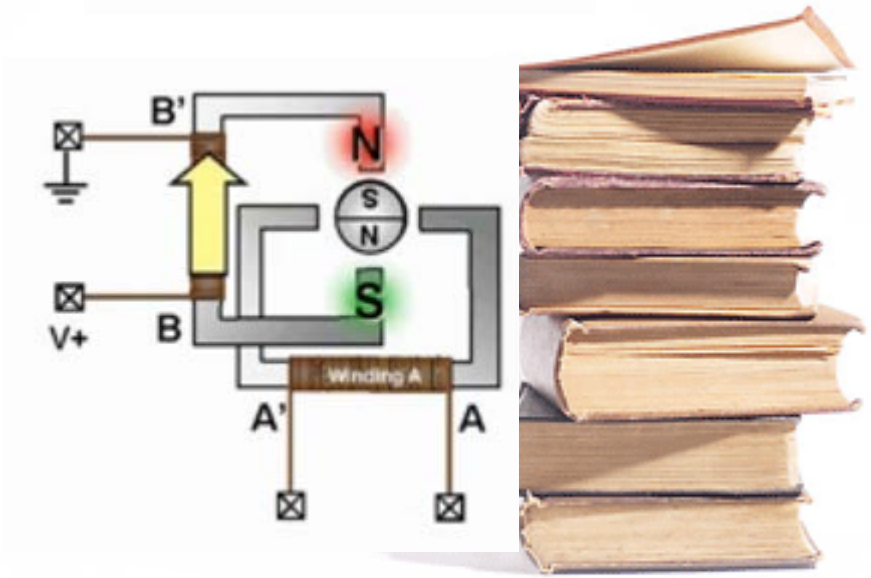

N
V+
OPEN
S
OPEN
OPEN
V+

Bipolar

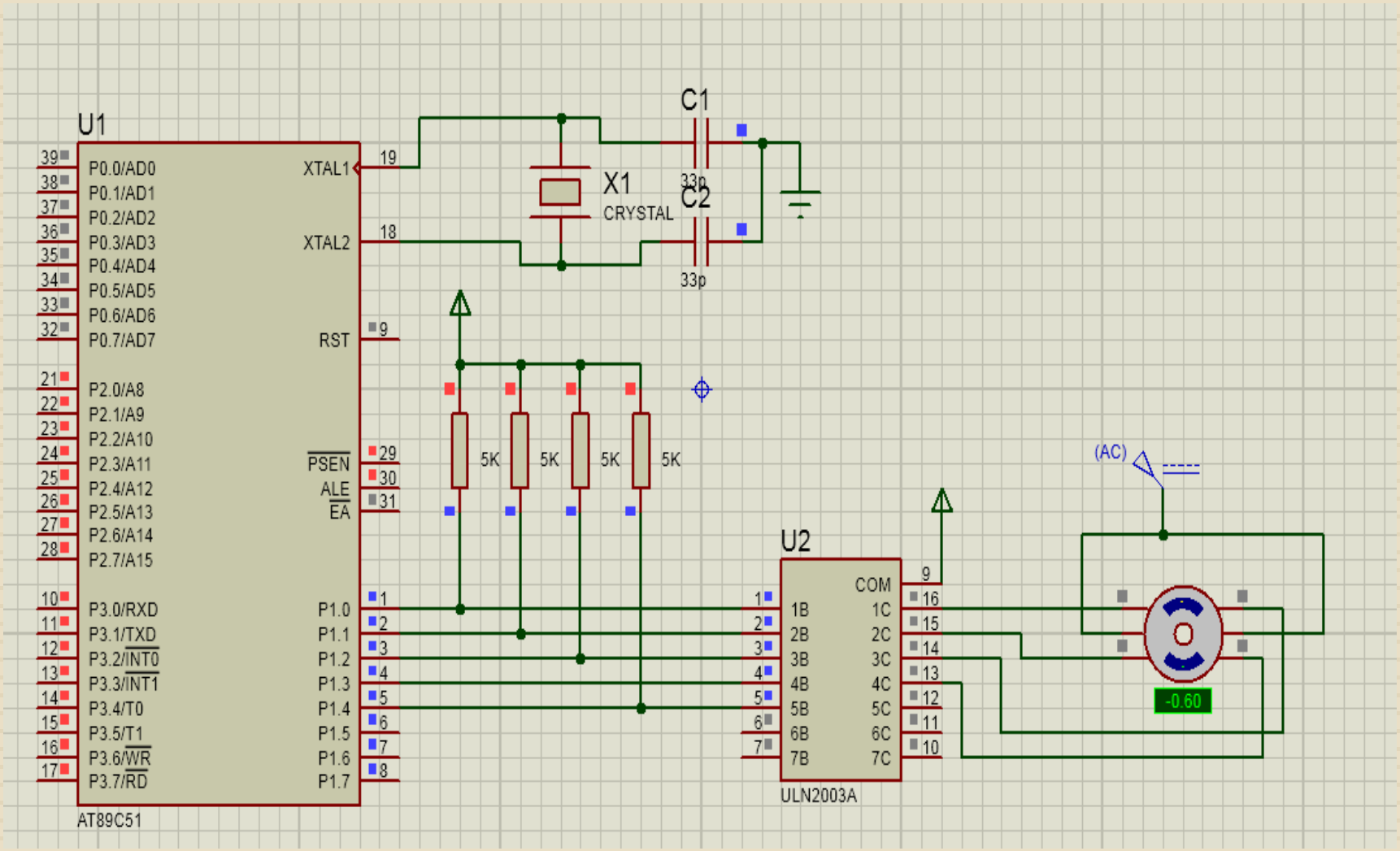

B'
Winding B
S
N
N
S
B
A'
A
V+


V+
B'
S
N
S
N
B
Winding A
A'
A


B'
Winding B
N
S
N
S
B
A'
A
V+


B'
N
S
N
S
V+
B
Winding A
A'
A

CIRCUIT DIAGRAM



EXCITATION SEQUENCE

FSE(CLOCKWISE)

STEP	Wa	Wb	Wa’	Wb’
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

DFSE(CLOCKWISE)

STEP	Wa	Wb	Wa’	Wb’
1	1	0	0	1
2	1	1	0	0
3	0	1	1	0
4	0	0	1	1

HSE(CLOCKWISE)

STEP	Wa	Wb	Wa’	Wb’
1	1	0	0	1
2	1	0	0	0
3	1	1	0	0
4	0	1	0	0
5	0	1	1	0
6	0	0	1	0
7	0	0	1	1
8	0	0	0	1

FEW TERMINOLOGIES:

- ## STEP ANGLE

θ=360/Steps Per Revolution

Steps Per Second=rpm*steps Per Revolution/60

- ## HOLDING TORQUE(Kg-cm)

Measured with rated voltage and rated current applied to motor

- **WRITE A PROGRAM TO ROTATE A MOTOR 64deg IN CLKWISE DIRECTION.MOTOR HAS A STEP ANGLE OF 2deg. USE DFSE SCHEME**

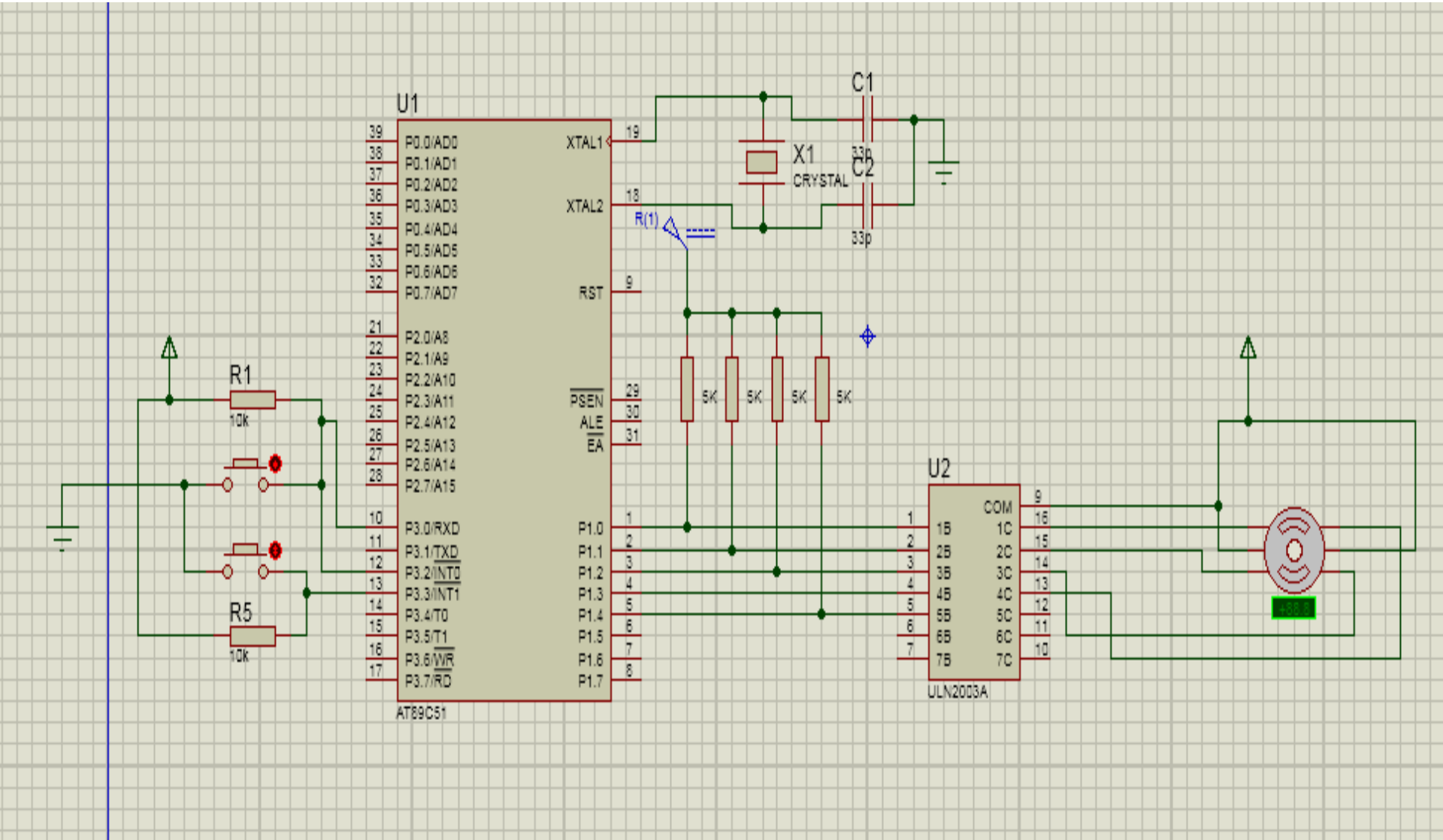
SOLN:steps per revolution=180

Movement per 4-step sequence=8deg

To move rotor 64deg eight consecutive 4-step sequences are required

```
ORG     0H
        MOV     A,#99H
        MOV     R0,#32
BACK:   RR      A
        MOV     P1,A
        ACALL   DELAY
        DJNZ    R0,BACK
END
```

Circuit diagram 2



PROGRAM2:

Write a program to rotate stepper motor in clkwise dir when INT0 is generated and in aclkwise dir if INT1 is generated.

```
ORG         0H
            LJMP        MAIN
ORG         30H
MAIN:
            MOV         IE,#10000013
HERE2:      JB          P3.2,HERE2
BACK1:      PUSH        82H
            PUSH        83H
            MOV         R5,#04H
BACK:       CLR A
            MOVC        A,@A+DPTR
            MOV         P1,A
            LCALL       DELAY
            INC         DPTR
            DJNZ        R5,BACK
            POP         83H
            POP         82H
            SJMP        BACK1
DELAY:
            MOV         TMOD,#01H
            MOV         TL0,#00H
            MOV         TH0,#00H
            SETB        TR0
HERE5:      JNB         TF0,HERE5
            CLR         TR0
            CLR         TF0
RET


ORG         0003H
            MOV         DPTR,#CLK
RETI

ORG         0013H
            MOV         DPTR,#ACLK
RETI

ORG         100H
            CLK: DB 09H,0CH,06H,03H
            ACLK:DB 03H,06H,0CH,09H

END
```

QUERIES?

THANK YOU